Playing With Fire: Dissecting Malicious Software

Other

The Quarterly Magazine for Digital Forensics Practitioners

DIGITAL FORENSICS / MAGAZINE

COMPETITION!
Win 3 brand new books from Syngress

ISSUE 03
1 MAY 2010

INSIDE

READING BETWEEN THE LINES
SPAM BEWARE!

Dr Tim Watson shows us how to perform forensic analysis on email headers

9 772042 061103 02
Issue 3 / £11.99 TR Media

PLAYING WITH FIRE: DISSECTING MALICIOUS SOFTWARE

CURRENT AND NEW TRENDS IN ANALYSING MALWARE BEHAVIOUR

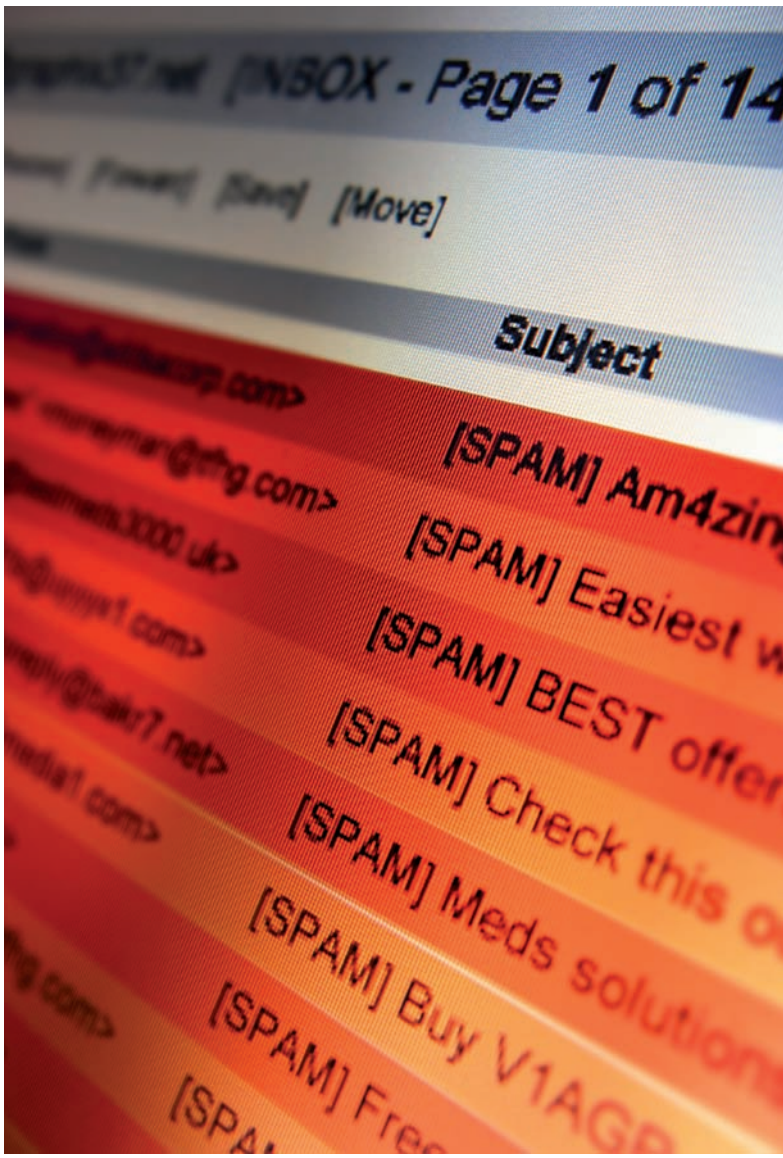
Modern malware is more sophisticated than it used to be and can easily mislead the investigator

by Ian Kennedy

/ EXPERT

Mention the word 'malware' in a word association game and few people would think to respond with 'weapon'. Malware is nearly always a means to an end in a much bigger picture. This could be the sale of information obtained, access to the compromised system or even the denial of access for the right price. Visualising the computer as the battlefield[1] and a network or computer system as a region or country then a malware attack becomes an offensive campaign against targeted systems. Continuing with this analogy, strategic decisions relating to how the campaign is fought become the overall design and execution of a malware attack. Decisions about how individual battles are fought are tactical in nature and equate to the techniques used in the construction and execution of malware tasks. In the midst of this are the forensic practitioners and security researchers. Their job is to be the weapons analysts and to reverse engineer these virulent and at times quasi-conscious weapons to understand their capabilities and behaviour.

It is difficult to imagine undertaking any offensive campaign without a range of tactical weapons, each suited to different tasks. The attacker can use a full selection of arsenal including



/ CAN I USE VMWARE TO STUDY MALWARE?

Sure, but be mindful of the fact that much of today's malware is emulator and debugger aware. Typically, malware inspects the environment where it is about to unpack and checks for signs of a non-native environment. The first sign of something suspicious and it invariably exits immediately without unpacking its code. Turning this around, this has a possible benefit for the user as it suggests you can install tools such as an emulator and a debugger to prevent malware from running. Longer term though, as virtualization becomes more common on a desktop it is likely that such environmental checks will no longer be a valid indicator of an instrumented system.

keyloggers, screen loggers, email redirectors, web Trojans, hostname lookup attacks, proxy attacks and rootkits to name a few. Each is suited to a different objective and each will often contain its own counter-detection measures. To start your campaign you need a weapons supplier.

/ DESCENDING UNDERGROUND

Buying weapons on the Black Market is not new. They are there to serve your every need, for a price. Recently appearing in the news[2], Zeus is an example of a DIY kit for building your own customised malware. With your freshly built malware it's not enough to simply locate it on a couple of websites and hope for passing surfers to get infected. You need to get it distributed to machines with identified vulnerabilities that can begin making you money quickly. That's where an Exploitation Pack comes in.

You can expect to pay around $100-250 to get your customised malware installed onto around 1,000 machines in the UK. Three widely used systems are Fiesta, Firepack and Sploit[3]. Now you need somewhere to store all your harvested data and manage your malware distribution. Anonymous 'bulletproof' servers offer a variety of packages and typically cost around $150 per month for hosting, with discounts for larger quantities.

/ ENTER THE WEAPONS ANALYST

With my practitioner hat on, I am principally motivated by the need to determine the events leading up to a limited set of circumstances when analyzing malware. I need to determine if the suspicious binary discovered on a computer is the cause of a given activity. There are two broad approaches taken to address this question of causality: static and dynamic analysis.

BUYING WEAPONS ON THE BLACK MARKET IS NOT NEW. THEY ARE THERE TO SERVE YOUR EVERY NEED, FOR A PRICE

In the static world the malware is lifeless, but not completely harmless as careless handling can be problematic. We must also remember though, that even our tools can be mislead in this exercise by the tactics of the malware author, leading to differing results between tools. Hence, as with computer

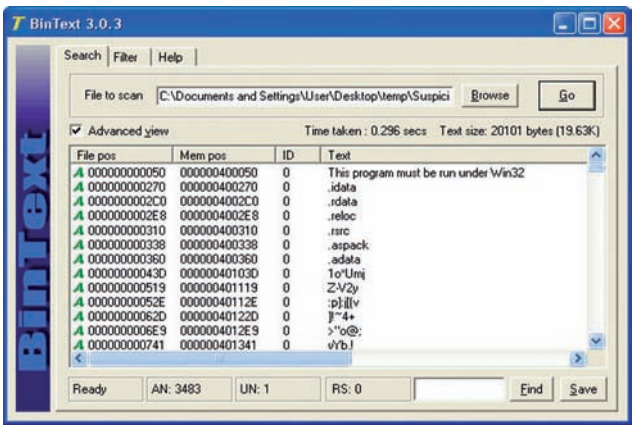


Figure 1

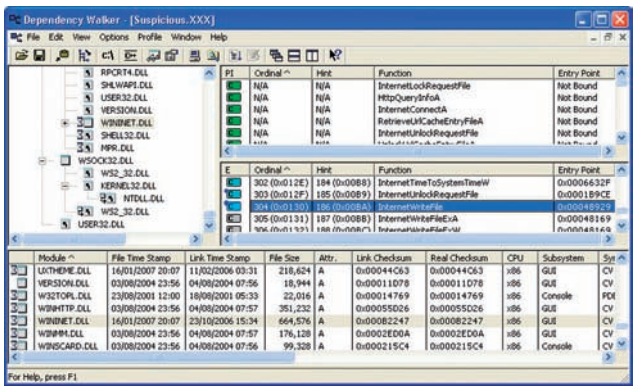


Figure 2

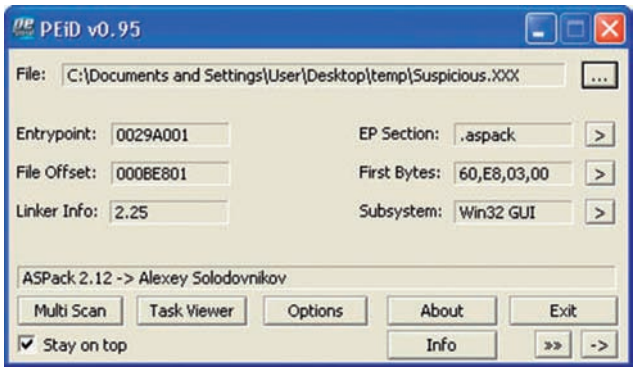


Figure 3

forensics, cross checking with more than one tool can increase confidence in what you are seeing is accurate. With this knowledge, we can safely commence our autopsy and dissect the specimen.

Online scanners are freely available to help us rapidly scan our suspicious binary in an attempt to identify it. Despite the ease of doing this, we have to stop and consider any legal issues involved, such as privacy and cross-border movement of data. This is especially important in the case of sensitive and government clients for whom we are conducting the investigation. With any legal issues resolved, we can don our gloves and surgical mask and submit our suspicious binary (previously identified by our AV scanner unhelpfully as simply 'Worm.Win32.Gen') to online scanner sites such as VirusTotal[4] and Jotti's malware scanner[5].

It is important to remember that a vendor's online scanner tool operates differently to its desktop product and so the precision is not the same. This becomes apparent when we examine how such online tools have identified the binary. The lack of consensus on even the name of the binary may give rise to doubts of what we are dealing with.

Not defeated, we turn to our preferred hex editor tools and examine the 'Magic Number' of this file; we see that it is 'MZP'. This indicates that the file is an executable file produced with Delphi. Knowing that a typical executable file contains varying amounts of string-based data we next apply that ever-useful Unix tool called 'strings' which has been ported to run in a Windows environment [6]. The 'BinText' tool[7] offers powerful string filtering and searching options. It will also denote both the file and memory offset of a string. Examining the output

from BinText we see the text '.aspack', see Fig 1. This suggests the file may be packed using the compression tool ASPack[8]. Sure enough, further on in the file (beyond that shown in Figure 1), there is little readable text, aside from references to API function names such as 'URLDownloadToFileA'. Tackling this problem by attempting to unpack the code is covered shortly.

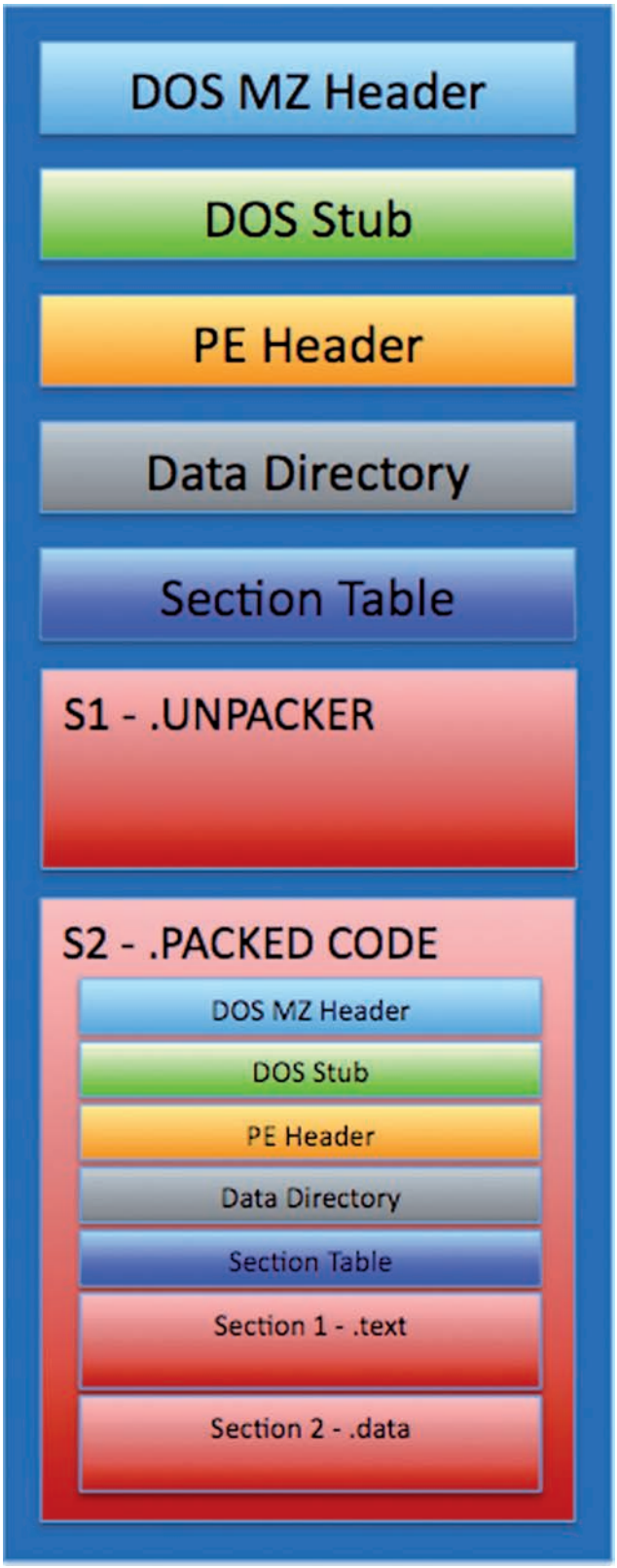


Figure 4

Just before we try to manipulate the code to unpack it, we may find it useful to examine its module dependencies. Malware will typically utilize the victim's own Windows API code to achieve its goals. The 'Dependency Walker'[9] allows us to examine which libraries and functions are called by the binary. Fig 2 indicates that our suspicious binary accesses 'WININET.DLL' which provides Internet functionality. One such function called is the 'InternetWriteFile'.

/ STEALTHY MALWARE

Malware writers are crafty. Like any military organisation, secrecy is key to protecting your assets. To this end, a variety of obfuscation techniques are used to try and prevent prying eyes from seeing what's happening under the hood. The first of these are packers.

MALWARE WRITERS ARE CRAFTY. LIKE ANY MILITARY ORGANISATION, SECRECY IS KEY TO PROTECTING YOUR ASSETS

Most packers operate only on Windows portable executable (PE) files and Dynamic Link Libraries (DLLs). Packers come in four broad varieties. The first of these, 'crypters', will gladly encrypt a malware file's content such that it will only decrypt portions of the malware in memory at any given time, frustrating any efforts to gain a full and unencrypted sample for analysis. Not surprisingly, crypters such as Yoda's Crypter[10] and PolyCrypt PE[11] are popular choices for obfuscating code[12]. This technique means that host-based detection technologies cannot inspect the binary prior to it being loaded from disk into memory. The second type of packer is the more traditional 'compressor' such as UPX. These do little more than compress the file's content without such trickery. Thirdly, 'bundlers' such as PEBundle[13] on the other hand, will produce a single packed file like a Zip container but will happily unpack files to RAM it needs to access, without extracting them to disk. Finally, 'protectors' such as Armadillo[14] and Temida[15] provide obfuscation through compression and licensing management capabilities for products found in the commercial arena.

Applying PEiD[16] to our suspicious binary, we identify the use of the packer 'ASPack 2.12' that our strings analysis suggested earlier, fig 3.

The typical packer will reorganise the headers in the file, rewrite the import table and set a new Original Entry Point (OEP). The end result is a PE binary containing within one of its sections a second, hidden binary image, like Russian dolls. When executed, the visible binary unpacks the inner binary, updates its own import table and finally makes a jump to the memory address of the unpacked binary. Thus, the malware is brought to life.

Disassemblers are programs designed to extract the assembly language of a target binary file and study the design and flow of the code. In our analogy, this gives us the blueprints for the weapon's behaviour. The tool of choice for many security researchers is IDA Pro[17]. Tools such as this

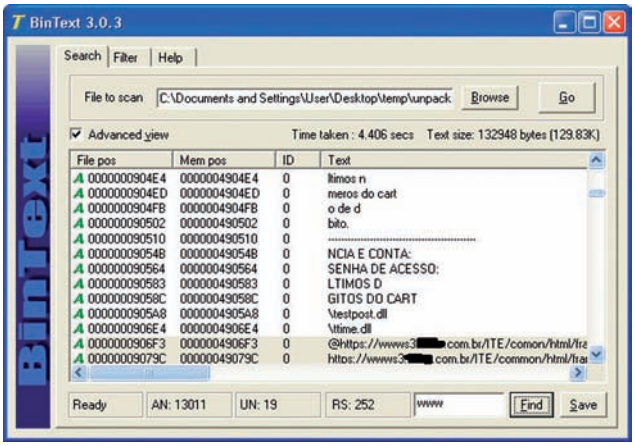


Figure 5

allow us to unpack (in some cases) and examine how the code would execute at the OpCode level, but require time and skill to use. Fortunately, there are tools available to unpack the packed code for us, enabling us to re-examine the strings and other resources used by the code.

Using 'AspackDie'[18] we can unpack the malware from its armour to a regular executable file. Loading this back into BinText we find references to online banking sites and the words 'conta', 'senha', 'gitos do cart' which (with the help of Google) in Portuguese mean 'account', 'password' and 'card digits', see Fig 5.

ANOTHER OBFUSCATION TECHNIQUE IS THE DESTRUCTION OF THE INTERRUPT ACCESS TABLE (IAT)

To avoid being unpacked, crypter packers have been known to insert 'junk' code when packing a binary to confuse a decompiler. Other tricks include redirecting or mutating original instructions with equivalent instruction codes to prevent memory dumps taking place. Even CRC checks are made by the extracted code to check the integrity of how it was extracted.

Given the challenges of static disassembly, you might think that simply running the code and monitoring its behaviour live would alleviate such problems. However, practitioners who seek to reverse engineer the code also face a relatively new and ingenious trick. Malware is now circulating that use VMProtectors[19]. This advanced technique effectively runs the malware code inside a mini virtual machine that is unpacked into memory first when the binary is executed. Each byte of the op-codes and operands forming the malware payload are randomly generated to form 'bytecode' when the malware is packed into the VMProtector. No two generated instances of the same malware will produce the same bytecodes. A practitioner running this type of malware will only see the virtual machine code and not the malware inside a debugger environment.

Another obfuscation technique is the destruction of the Interrupt Access Table (IAT). This table stores all the memory addresses where an executable may find the standard routines provided by the Windows Application Programming Interface (API). Malware employing this technique can avert detection by making calls to the API routines directly by memory address rather than name. In this way, the practitioner never sees the calls made to the API.

In moving to dynamic analysis, we are arming the weapon. In our analogy we are positioning the weapon in an artificial desert town then retreating to our viewing bunkers. Thus, not wanting to liberate malware on our forensic workstation we typically turn to the use of emulators to virtualize a machine, creating that artificial town in which to detonate our weapon. These days, however, malware is smarter than it used to be (more on this in a later) as before detonation it will check its environment is real. I guess those mannequin dummies in our artificial town are no substitute for real people.

/ WEAPONS CATALOGUE

Malware classification used to be simple. These days much of what is in the wild is a Darwinian blend of threats covering multiple attack vectors. Again returning to our military analogy, this is akin to a flanking offensive tactic. A blended threat might, for example, launch a Denial of Service (DoS), install a backdoor and overwrite local system files in one attack. Multiple delivery mechanisms could be used too to increase the success of gaining access. So a worm may arrive by both email and a file sharing mechanism. The following list, based upon definitions provided by VirusList[52] identifies the elements of these blended threats:

WORMS
Typically, do not require human interaction to operate and are classified by the propagation and/or installation method employed. Examples of these include email, Instant Messaging, IRC and peer-to-peer file sharing mechanisms.

VIRUSES
Generally require human interaction to be initiated and are classified according to their area of operation. This can be either the file system, boot sector, macro or scripting areas.

TROJANS
Classified by their action, this group of malware will typically act as the mechanism for delivery of some other item such as Backdoors, Droppers (unpacking it's payload), Downloader (sourcing it's payload from online), Proxies (using a victim to hide Internet activity) and Spies (to monitor keystrokes and screen activity).

DOS/DDOS
Rendering the victim machine to the role of a pawn, when instructed it obediently commences a repeated request to a designated server.

EXPLOITS
What you might consider 'opportunist' malware that is embedded with code targeted at unpatched systems containing vulnerabilities. A JPG file, for example, can contain code to exploit a buffer overrun when viewed53.

NUKERS
Exploiting vulnerabilities on the victim host, the system is attacked and made to cause fatal errors.

In this virtual realm, we install the monitoring tools of our choice. In doing so we seek to instrument the usual areas such as file, registry, network and process activity. Process Monitor[20] wraps up into one tool what used to split across several tools. Wireshark[21] allows us to monitor network traffic filtered by protocol. In analyzing malware we seek to emulate the Internet by setting up a second machine on our test LAN as a web server and using fake DNS. In doing this we illicit a response from the malware when the website addresses found during our static analysis are resolved via local DNS to our internal web server.

Debuggers are programs that allow us to execute a binary in a controlled fashion, tracing step-by-step the execution path followed. Given the malware is being executed in a restricted manner, it is still essential to do this step on an expendable and isolated machine. Executing our binary in the debugger OllyDbg[22] we are greeted with the message that the entry point is 'outside' the code area, confirming our suspicions that the OEP has been shifted.

A sophisticated defence tactic used by malware is a technique known as 'Dynamic code replacement'. A malware binary is executed and partially unpacks code that starts a second, benign process. The unpacked code then allocates memory within the domain of this new process, typically by calling the 'writeProcessMemory' API. The remainder of the malware's code is then unpacked into the domain of the new process. Finally, the entry point is then reset to point to the start of the malware code, completely skipping the logic of the benign process. This is like building the Trojan horse within the castle walls.

/ CURRENT RESEARCH TRENDS

Reaching beyond the confines of a specific investigation, the academic community is exploring a number of exciting approaches to malware analysis.

On a static level it is possible to compare suspicious binaries with known samples. Two groups[23] [24] borrow a concept from biology known as phylogenetics which studies the evolutionary relationship among various groups of organisms. The basic principle is to cluster calls made to similar or identical functions within a pair of binaries. This approach becomes ineffective for malware that obfuscates its code and calls to API functions.

In a move similar to that which helps trace stolen money, tainting[19] [25] [26] is a technique used to mark data, as it propagates through the execution paths of malware.

The hooking of both kernel and application level APIs are used[24] [26] [27] to monitor calls to API functions through callback functions. One project known as Mazalyzer[28] is built upon the Microsoft research project, Detours[29]. Their approach seeks to overcome the challenges of stealthy malware by monitoring for suspicious process startup activity. The creation of a second process is deemed suspicious as dynamic code injection is legitimately used in benign applications such as Skype[30], but does not normally attempt to start a second process.

/ EMULATION AND DEBUGGING

As a security researcher you seek to have a controlled environment to study your malware. There are a variety of software emulation technologies to choose or build upon such as CWSandbox[31], QEMU[32], User Mode Linux (UML)[33], LiveView[34], Norman[35], Bochs[36] and VMWare[37]. Firing up your malware in one of these virtual environments allows you to both manipulate the inputs given to the malware and instrument its behaviour while filtering out unwanted noise from other system activity. On their own though, these emulation tools are not enough to defeat most malware and at times may be unsuitable as the malware detects their presence.

If you consider the deployment of malware as the use of a weapon, then the effectiveness of that weapon is in jeopardy if your enemy discovers how your weapon operates and where its weaknesses lie. Thus intelligence is a critical element to the effectiveness of your weapon. So one of the typical behavioural characteristics of malware is reconnaissance. Before a victim's machine is probed to identify vulnerabilities, a separate 'reccy' is run to look for the telltale signs of an emulated environment. These include manufacturer device identifier strings such as 'vmnet1', which is reported for systems hosting VMWare containing a network interface. A typical response to such a discovery is to exit the process immediately. More sophisticated binaries will only unpack enough code to make these checks first to check it's safe to come out of hiding in the packed file. The ScoobyNG[38] tool uses shell scripts to identify vendor strings.

THE EFFECTIVENESS OF THAT WEAPON IS IN JEOPARDY IF YOUR ENEMY DISCOVERS HOW YOUR WEAPON OPERATES

Sticking with VMWare as an example, other clues include the presence of the key 'vmscsi' in the Windows registry and driver files such as 'buslogic.sys and 'vmscsi.sys' on the disk.

Emulators and debuggers will also make subtle changes to the kernel memory layout of the emulated machine. Thus a simple check of the address of a standard API routine such as 'createProcess' will indicate the system is not running on native hardware. Even the titles of open windows and live processes can be checked.

/ MALWARE FORENSICS

Malware Forensics: Investigating and Analyzing Malicious Code
Authors: James Aquilina, Eoghan Casey, Cameron Malin
Publisher: Syngress
ISBN-10: 159749268X
ISBN-13: 978-1597492683

This book details methodologies for the collection and analysis of volatile data and artifacts in the context of a malware investigation. Practical in its approach, tools are cited throughout the book along with a discussion of the legal issues involved, which include UK and cross-border issues, which is critical given the nature of malware. The book forms an excellent starting point from which to study malware before moving on to more comprehensive, multiple path methodologies.

A common side effect of using an emulator is that certain CPU operations are not fully supported in the emulated environment[39 40]. It is of little surprise then that malware authors use this to their advantage as an indicator of an emulated system. Some malware will even measure the speed of execution of certain instructions to check for emulators and debuggers[39].

To tackle these problem hardware emulators such as JoeBox[41] and Ether[42] (which both use emulation techniques based upon Intel-VT) have been developed. Tools based upon Ether have outperformed the unpacking tools Renovo and PolyUnpack as well as the sandbox tools Anubis and Norman.

/ MULTIPLE PATH ANALYSIS

ALMOST ALL EXECUTABLE CODE WILL MAKE DECISIONS DIRECTING PROGRAM FLOW DOWN ONE BRANCH OR ANOTHER

Executing a malware binary to perform dynamic analysis, be it on an emulated or native machine, present the practitioner and security researcher with another problem: how do we know we have seen the entire capabilities of the binary? Almost all executable code will make decisions directing program flow down one branch or another. Executing a temporal virus such as Conficker.B will behave differently both before and after a given date.

To address this, approaches have been developed[25 43] that allow a malware sample to be executed and halted when a decision instruction is encountered. A snapshot is then taken of the system and the process is allowed to continue. When the branch is exhausted (or after a sufficient time has passed), the system is rolled back to the snapshot and the data manipulated so that another branch is explored.

/ ONLINE ANALYSIS

Some research projects have developed into online analysis tools available for use by practitioners and researchers alike. Going beyond a simple virus scan of a submitted sample, these tools virtualize a given sample and monitor its behaviour.

Anubis[44] (developed from TTAnalyze[45]) and CWSandbox are well-established projects. Commercial solutions include ThreatExpert[46] (a tool maintained by PCTools) and Norman Sandbox[35].

/ CONCLUSION

As we emerge from this dark underworld, we must reflect on the impact of our findings. There is much information that can be gleaned from the tools at our disposal. However, for the forensic practitioner there is on balance, more we do not know about a given sample of malware than what we do know. It is simply not enough to change a few conditions and run the malware, as this typically will not induce the malware to perform to it's full capability. Even for the security researcher, equipped with the skills and time to reverse engineer code, it is a huge challenge to achieve a full understanding of a malware binary employing new obfuscation techniques, in the short time frame between arrest and charge of a suspect in a criminal investigation. It seems that modern malware not only comes in sophisticated armored tanks these days, but even checks nobody's watching before firing. /

REFERENCES

1. Fernandez J, M., Bureau P. Optimising malware. Conference Proceedings of the IEEE International Performance, Computing, and Communications Conference 2006 01/01;2006:577-86.
2. BBC News : Two held in global PC fraud probe [Internet] [cited 2009 11/21/2009]. Available from: http://news.bbc.co.uk/1/hi/england/manchester/8366504.stm.
3. Erasmus J. Anatomy of a malware attack. Network Security 2009 1;2009(1):4-7.
4. VirusTotal - Free Online Virus and Malware Scan [Internet] [cited 2009 11/17/2009]. Available from: http://www.virustotal.com/.
5. Jotti's malware scan [Internet] [cited 2009 11/17/2009]. Available from: http://virusscan.jotti.org/en.
6. Sysinternals utility - Strings [Internet] [cited 2009 11/22/2009]. Available from: http://technet.microsoft.com/en-gb/sysinternals/bb897439.aspx.
7. Foundstone Free Tools [Internet] [cited 2009 11/28/2009]. Available from: http://www.foundstone.com/us/resources/proddesc/bintext.htm.
8. ASPACK SOFTWARE - Best Choice Compression and Protection Tools for Software Developers [Internet] [cited 2009 11/28/2009]. Available from: http://www.aspack.com/.
9. Dependency Walker (depends.exe) Home Page [Internet] [cited 2009 11/28/2009]. Available from: http://www.dependencywalker.com/.
10. Yoda's Crypter [Internet] [cited 2009 11/17]. Available from: http://yodap.sourceforge.net/.
11. PolyCrypt PE [Internet] [cited 2009 11/17/2009]. Available from: http://www.jlabsoftware.com/.
12. Yan W, Zhang Z, Ansari N. Revealing packed malware. IEEE Security & Privacy 2008 09/01;6(5):65-9.
13. PECompact Executable Compressor [Internet] [cited 2009 11/17/2009]. Available from: http://www.bitsum.com/pecompact.php.
14. Silicon Realms / SoftwarePassport / Armadillo - The Home of SoftwarePassport [Internet] [cited 2009 11/17/2009]. Available from: http://www.siliconrealms.com/index.html.
15. Oreans Technology : Software Security Defined. [Internet] [cited 2009 11/17/2009]. Available from: http://www.oreans.com/.
16. PEiD - Packer, Crypter and Compiler detection [Internet]. Available from: http://www.peid.info/.
17. IDA Pro Disassembler - multi-processor, windows hosted disassembler and debugger [Internet] [cited 2009 11/22/2009]. Available from: http://www.hex-rays.com/idapro/.
18. Aaron's Homepage- include AspackDie [Internet] [cited 2009 11/28/2009]. Available from: http://209.85.229.132/search?q=cache:zOGmZUZCz2wJ:www.exetools.com/unpackers.htm+aspackdie&cd=2&hl=en&ct=clnk&gl=uk.
19. Sharif M, Lanzi A, Giffin J, Wenke Lee. Automatic reverse engineering of malware emulators. Security and Privacy, 2009 30th IEEE Symposium on 2009:94-109.
20. Process Monitor [Internet] [cited 2009 11/28/2009]. Available from: http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx.
21. Wireshark · Go deep. [Internet] [cited 2009 11/28/2009]. Available from: http://www.wireshark.org/.

22. OllyDbg v1.10 [Internet] [cited 2009 11/22/2009]. Available from: http://www.ollydbg.de.
23. Digital genome mapping–advanced binary malware analysis. Virus bulletin conference; 2004. .
24. Wagener G, State R, Dulaunoy A. Malware behaviour analysis. Journal in Computer Virology 2008 11/01;4(4):279-87.
25. Moser A, Kruegel C, Kirda E. Exploring multiple execution paths for malware analysis. Proceedings - IEEE Symposium on Security and Privacy 2007 01/01:231-45.
26. Scalable, behavior-based malware clustering. Network and distributed system security symposium (NDSS); 2009. .
27. Inoue D, Yoshioka K, Eto M, Hoshizawa Y, Nakao K. Malware behavior analysis in isolated miniature network for revealing malware's network activity. IEEE International Conference on Communications 2008 01/01:1715-21.
28. Liu L, Chen S. Malyzer: Defeating anti-detection for application-level malware analysis; applied cryptography and network security, 7th international conference, ACNS 2009, paris-rocquencourt, france, june 2-5, 2009. proceedings. 2009;5536:201-18.
29. Detours - Microsoft Research [Internet] [cited 2009 11/22/2009]. Available from: http://research.microsoft.com/en-us/projects/detours/.
30. Skype uncovered [Internet] [cited 2009 11/18]. Available from: http://www.ossir.org/windows/supports/2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf.
31. CWSandbox [Internet] [cited 2009 11/22/2009]. Available from: http://www.cwsandbox.org/.
32. QEMU [Internet] [cited 2009 11/22/2009]. Available from: http://www.nongnu.org/qemu/about.html.
33. Dike J. User mode linux Upper Saddle River, NJ: Prentice Hall; 2006. id: 1; change_time="2007-06-28T03:04:24Z" price_time="2009-10-04T19:36:27Z" edition_info="(pbk. : alk. paper)" language="eng" physical_description_text="p. cm." lcc_number="QA76.76" dewey_decimal_normalized="5.432" dewey_decimal="005.4/32"; Includes bibliographical references and index.; .
34. Live View [Internet] [cited 2009 11/22/2009]. Available from: http://liveview.sourceforge.net/.
35. Norman | Norman SandBox® [Internet] [cited 2009 11/22/2009]. Available from: http://www.norman.com/technology/norman_sandbox/en-us.
36. bochs: The Open Source IA-32 Emulation Project (Home Page) [Internet] [cited 2009 11/22/2009]. Available from: http://bochs.sourceforge.net/.
37. VMware Business Infrastructure Virtualization: Beyond Virtual Machines & Servers [Internet] [cited 2009 11/22/2009]. Available from: http://www.vmware.com/.
38. ScoopyNG [Internet] [cited 2009 11/22/2009]. Available from: http://www.trapkit.de/research/vmm/scoopyng/index.html.
39. Chen X, Andersen J, Mao ZM, Bailey M, Nazario J. Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware. Dependable Systems and Networks with FTCS and DCC, 2008 DSN 2008 IEEE International Conference on 2008 06/24:177.
40. Testing CPU emulators. Proceedings of the eighteenth international symposium on software testing and analysis - ISSTA '09; 2009. id: 1; isbn: print 9781605583389; publication_type: full_text.
41. Joebox a secure Sandbox Application for Windows to analyse the Behaviour of Malware [Internet] [cited 2009 11/22/2009]. Available from: http://joebox.org/#.
42. Ether: Malware analysis via hardware virtualization extensions. Proceedings of the 15th ACM conference on computer and communications security - CCS '08; 2008. id: 1; isbn: print 9781595938107; publication_type: full_text.
43. Purui S, Lingyun Y, Dengguo F. Exploring malware behaviors based on environment constitution. Proceedings - 2008 International Conference on Computational Intelligence and Security, CIS 2008 2008 01/01;1:320-5.
44. A View on Current Malware Behaviors [Internet] [cited 2009 11/5/2009]. Available from: http://www.usenix.org/events/leet09/tech/full_papers/bayer/bayer_html/.
45. Bayer U, Kruegel C, Kirda E. TTAnalyze: A tool for analyzing malware 2006.
46. ThreatExpert - Submit Your Sample Online [Internet] [cited 2009 11/22/2009]. Available from: http://www.threatexpert.com/submit.aspx.
47. BBC NEWS | Technology | Millions tricked by 'scareware' [Internet] [cited 2009 11/29/2009]. Available from: http://news.bbc.co.uk/1/hi/8313678.stm.
48. APWG Phising Activity Trends Report - First Half 2009 [Internet]. Available from: http://www.antiphishing.org/reports/apwg_report_h1_2009.pdf.
49. Websense Security Labs report - State of Internet Security, Q1-Q2 2009 - Websense Features [Internet] [cited 2009 11/29/2009]. Available from: http://community.websense.com/blogs/websense-features/archive/2009/09/15/websense-security-labs-report-state-of-internet-security-q1-q2-2009.aspx.
50. China accused over global computer spy ring | World news | The Guardian [Internet] [cited 2009 11/22/2009]. Available from: http://www.guardian.co.uk/world/2009/mar/30/china-dalai-lama-spying-computers.
51. Germany accuses China of industrial espionage | World news | The Guardian [Internet] [cited 2009 11/22/2009]. Available from: http://www.guardian.co.uk/world/2009/jul/22/germany-china-industrial-espionage.
52. Viruslist.com - Information About Viruses, Hackers and Spam [Internet] [cited 2009 11/29/2009]. Available from: http://www.viruslist.com/.
53. Microsoft Security Bulletin MS04-028: Buffer Overrun in JPEG Processing (GDI+) Could Allow Code Execution (833987) [Internet] [cited 2009 11/29/2009]. Available from: http://www.microsoft.com/technet/security/bulletin/ms04-028.mspx.

AUTHOR BIO

Ian Kennedy is employed as a Consultant by Control Risks. Operating both in and out of his base in London, Ian has undertaken forensic investigative work in a variety of jurisdictions around the world. With a background in computer forensics within law enforcement and C++ programming, Ian continues to take an active interest in contributing to both further and higher education programmes through his various roles as an Associate Lecturer for the Open University, his visiting lecturers given to a number of Universities and his work as a part-time tutor for a local further education college. Currently following a Doctorate in malware research, his research interests focus on malware behavior.

Although the example is a very small one it illustrates the process of question and answer through the development of a model, rather than just the accumulation of data. This approach, although relevant to all forensic analysis is particularly important in psychosocial forensics where there is not a lot of data to collect by comparison to the digital case. The value of a model-based approach is that at the end of the analysis there is something, the model, to go forward to the next analysis, which can provide a jumping off point for that analysis. This leads to a possible new step in the forensic process, when viable and relevant models are present, that of model validation that is not normally part of a forensic investigation.

We have taken a psychosocial example to model in this paper rather than a digital or physical one but the approach is equally applicable to those other two areas. The choice of a psychosocial context was to explicitly show that the approach is not tied to the technical aspects of forensic analysis.

Figure 5 and Figure 6 show the relevance of security zones in relation to Petri Net models. They show the boundary and context of the various agents concerns and control.

DISCOVERING THESE RELATIONSHIPS CAN SUGGEST FORMS OF INFORMATION TO INVESTIGATE THAT WOULD NOT OTHERWISE HAVE BEEN INVESTIGATED

In Figure 7 we see a simple conceptual model version of the insider threat Petri Net model above. If we link this to the other conceptual models such as those shown in part 1, we can see where precursor behaviour might leave manifest evidence of its presence. From this the nature of the targeted monitoring can be determined along with any forensic activity this could involve.

One of the features of ORM is its ability to be verbalized and so be communicated to others who may not comprehend the formal models. Thus Figure 7 can be rendered in the following verbal form:

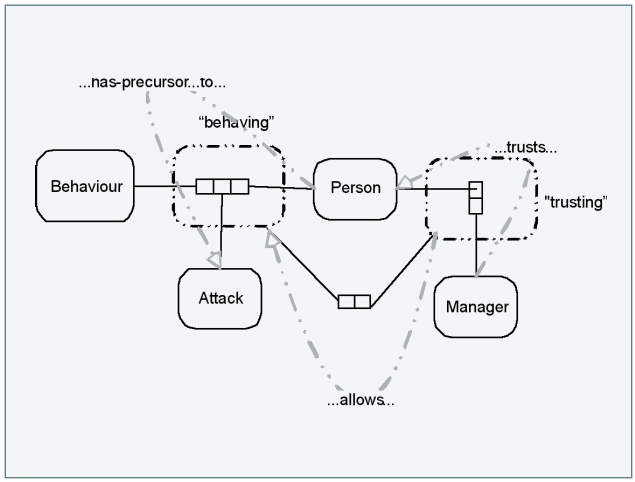


Figure 6. Targeted Monitoring Countermeasures

A Person presents precursor Behaviour in relation to an Attack. Managers trust Persons. The trusting allows this precursor behaving to exist.

How do we use these conceptual models for forensic work? Again as in the case of Petri Nets we are looking for causes. The answer is to follow the relationship concept chains, just as in the Petri Nets case we follow the condition action chains. So for example, we can go from behaviour to data on hard disk via the documents the person has used. Discovering these relationships can suggest forms of information to investigate that would not otherwise have been investigated. Conceptual models are able to provide models other than just for the logical perspective and thus enable forensic analysis to move consistently between the different perspectives as required. This is particularly relevant to operational forensics and its need to cover more than the logical i.e. digital, aspects of an incident. To get the full benefit the conceptual model needs incorporating into the net model.

/ SUMMARY

In these two parts we have introduced a number of modelling methods that could be used to guide aspects of forensic analysis, as well as suggesting a new arena of forensic analysis. The use of Petri nets in this area is not new as can be seen by

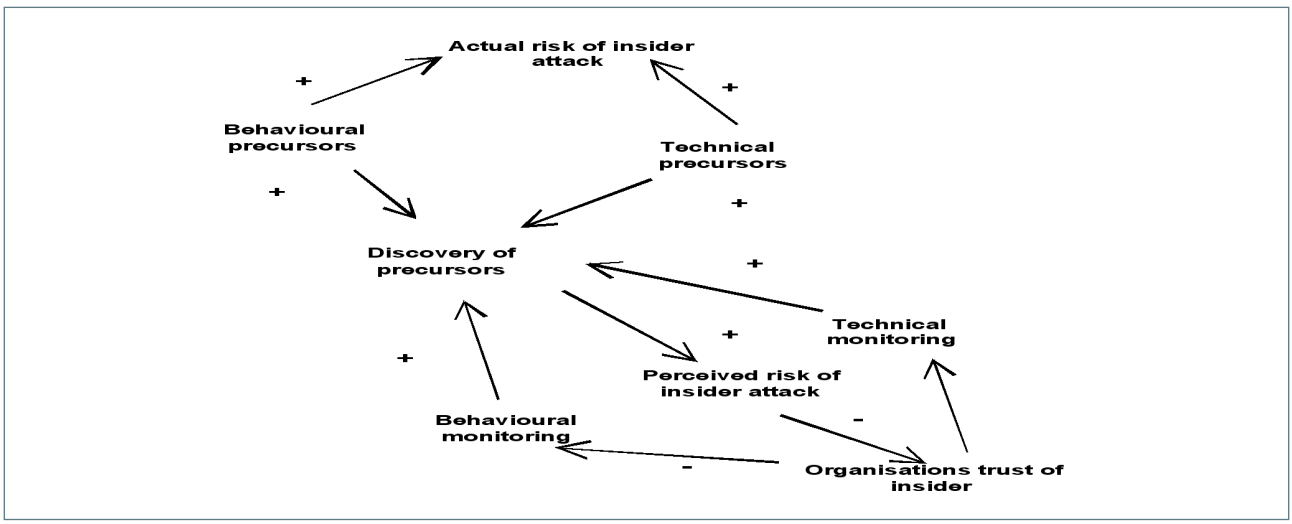


Figure 7. Conceptual Model of Attack Precursor Behaviour